Embedded Linux License Compliance for Hackers & Makers

Paul Barker

Konsulko Group

FOSDEM 2021

About Me

- Involved in Yocto Project since 2013
- Work across the whole embedded stack
- Principal Engineer @ Konsulko Group
- https://www.konsulko.com/

Contact Details

- Email: pbarker@konsulko.com
- Web: https://pbarker.dev/
- Twitter: https://twitter.com/pbarker_dev
- YouTube: https://www.youtube.com/channel/UCvnVQTiuS9-1dxZI-SJGBRA

Disclaimer

- IANAL
- This presentation is not legal advice
- Best practices are given based on my experience as a developer and an open source community member
- If in doubt, consult an appropriate lawyer

Introduction

- Lots of information and tools available for open source license compliance
- Not well targeted for hobbyists, individual makers and small businesses distributing devices containing open source software in small volumes
  - Complex tools
  - Time & effort consuming methods

Why care

- For corporations the aim of license compliance is likely to reduce legal risk and to gain influence in relevant open source communities
- For hackers & makers the priorities are likely to be different
  - Empowering users
  - Being a good citizen of the free software & open source movements
- Capturing source code & build scripts helps reproducibility of builds
  - Sources do often disappear off the internet

What are you distributing?

- Physical device with open source software installed
  - Let’s assume the recipient has internet access
- Software image for download from a website
  - Containing kernel, bootloader, rootfs, etc; not just a single software package
- It doesn’t matter if any price is charged
- In a small business, you can ignore distribution to other workers as part of your job

Common license conditions

- Provide license text and notices (BSD, MIT, etc)
  - On device?
  - In documentation?
  - On website?
- Provide Complete Corresponding Source (GPL)
  - Published directly?
  - Via an offer letter?

General guidelines

- Use an embedded Linux build system
  - Buildroot
  - OpenEmbedded/Yocto Project
  - etc
  - These systems help collect license text & source code as needed
- Avoid modifying the software image in a post-build script
- Avoid adding additional software during manufacturing test processes

Things to avoid

- Desktop/server distros
- OpenWRT
- Pulling images from Docker Hub and similar container registries
- Building container images with a Dockerfile
- Why?
  - Difficult to collect license text
  - Difficult to collect source code of copyleft packages

Things to use carefully

- Pre-compiled toolchains
  - E.g. ARM toolchain
  - Libraries from the toolchain typically end up in the distributed image
  - Ensure source code is collected
- Language-specific package managers
  - E.g. NPM, Cargo, etc
  - May not offer easy ways to collect license text or correct source code
- Un-reviewed third-party Makefiles
  - Watch out for downloads or use of online tools during build

Publishing license text & notices

- Format text and notices into a HTML or TXT page and include in the software image, accessible from a UI if possible
- An alternative:
  - License text & notices can easily be collected in a git repository
  - Update with a new commit for each distributed software release
  - Take advantage of free git repository hosting
  - Distribute a link to this with your product

Publishing source code

- Publish sources via a cheap online file host
  - Backblaze B2 + CloudFlare (https://www.cloudflare.com/en-gb/bandwidth-alliance/backblaze/)
  - Hetzner storage boxes
  - etc
- Deduplicate between releases where possible
- Ensure any patches are included
  - Watch out for “hidden patches” (e.g. sed scripts, etc)
  - Ensure the patch order is recorded

Providing build scripts

- Don’t forget this one!
  - GPLv2 says to include “scripts used to control compilation and installation”
- Best to provide sources for the build system
  - Buildroot repository with any customisations
  - OpenEmbedded repositories plus all layers in use
- Ensure any local configuration is included if it’s not tracked in git

Testing

- Mistakes are easy to make, that’s why we have tests
- There is one gold standard test:
  - Can the image be recreated from the sources & build scripts you publish?
- Automate this test if possible!
- Run it on every release

Using Buildroot

- Run `make legal-info`
  - Less configurable than the tools provided by OpenEmbedded/Yocto Project but it’s well documented and easy to use
  - Captures original sources, patches and license text
- Also see the talk “License compliance for embedded Linux devices with Buildroot” by Luca Ceresoli at FOSDEM 2020

Using OpenEmbedded/Yocto Project

- Enable the archiver bbclass
    - Alternatively archive the downloads directory but this is less flexible
- Archive deployed licenses directory or enable installation of license text into the target image
- See my previous talks:
    - “License Compliance in Embedded Linux with the Yocto Project” at Embedded Linux Conference Europe 2019
    - “Open Source License Compliance with Yocto Project” at Linaro Virtual Connect 2020

Other relevant projects

- REUSE: https://reuse.software/
- Openchain: https://www.openchainproject.org/
- OSS Review Toolkit: https://github.com/oss-review-toolkit/ort
- Software Heritage: https://www.softwareheritage.org/
- Fossology: https://www.fossology.org/

Open work

- Status of license compliance tools in
  - OpenWRT
  - PTXDist
  - Other build systems?
- Improving the state of language package managers
- Integrating with other projects & tools

Q&A